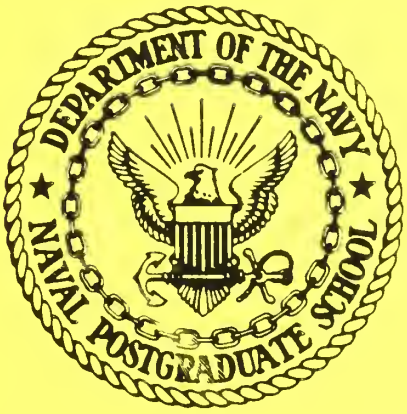
NPS54-85-014

NAVAL POSTGRADUATE SCHOOL

Monterey, California



A DECISION SUPPORT SYSTEM FOR THE NATIONAL COMMUNICATIONS SYSTEM

by

Norman R. Lyons

March 1986

Approved for public release. Distribution Unlimited.

Prepared for:
National Communications Systems
Arlington, VA 22204

NAVAL POSTGRADUATE SCHOOL
Monterey, California

RADM. R. H. Shumaker
Superintendent

David A. Schrady
Provost

The research summarized herein was sponsored by National Communications System.

This report was prepared by:

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION: Unclassified	1b RESTRICTIVE MARKINGS
SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION / AVAILABILITY OF REPORT: Distribution Unlimited
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S): NPS-54-85-014	5 MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION: Naval Postgraduate School; 6b OFFICE SYMBOL (If applicable): 54	7a NAME OF MONITORING ORGANIZATION: National Communications System
6c. ADDRESS (City, State, and ZIP Code): Monterey, CA 93943	7b ADDRESS (City, State, and ZIP Code): 8th Street and South Courthouse Road, Arlington, VA 22204
8a. NAME OF FUNDING / SPONSORING ORGANIZATION: NCS; 8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER: MIPR HC1001-6-10118
8c. ADDRESS (City, State, and ZIP Code): 8th. Street and South Courthouse Road, Arlington, VA 22204	10 SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO	PROJECT NO	TASK NO	WORK UNIT ACCESSION NO

11 TITLE (Include Security Classification): A Decision Support System for the National Communications System

12 PERSONAL AUTHOR(S): Norman R. Lyons

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Technical	FROM June 85 TO Dec 85	March 1986	41

16 SUPPLEMENTARY NOTATION

17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

20 DISTRIBUTION / AVAILABILITY OF ABSTRACT	21 ABSTRACT SECURITY CLASSIFICATION
☐ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	
NAME OF RESPONSIBLE INDIVIDUAL: Norman R. Lyons	22b TELEPHONE (Include Area Code): 408-646-2666; 22c OFFICE SYMBOL: 54Lb

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

A DECISION SUPPORT SYSTEM FOR THE NATIONAL COMMUNICATIONS SYSTEM

by

Norman R. Lyons

Administrative Sciences Department
Naval Postgraduate School
Monterey, California 93943
October 1985

SUMMARY

Problem

The Office of the Manager of the National Communication System (NCS) requires timely accurate information about communications resources during national disasters. This led to the development of a pair of decision support systems (DSS) for the National Emergency Management teams. These systems are the Emergency Preparedness Management Information System (EPMIS) and the Fly-Away Management System (FAMIS).

This report covers the FAMIS system. FAMIS currently exists as a prototype written in BASIC and implemented on portable Otrona microcomputers. The prototype is in an early stage of development. It has been useful for defining the final system.

Now that the concept has been developed, it is time to move beyond the prototype stage. NCS must start a disciplined development effort. The remainder of this report deals with the definition of the problem and the process of setting up a development effort.

Objective

There were several objectives in this project. They were:

1. Determine how to speed up the current FAMIS prototype.
2. Determine how to structure and integrate EPMIS and FAMIS.
3. Suggest approaches to long term FAMIS structure and design.
4. Suggest administrative approaches for EPMIS/FAMIS.

Approach

In performing this work, I did the following:

1. Analyzed FAMIS code.
2. Rewrote portions of FAMIS code.

3. Reviewed reports from NCS and Booz, Allen and Hamilton on the EPMIS/FAMIS system.

4. Reviewed related DSS and system development literature.

5. Reviewed computer software for useful packages.

Results and Conclusions

The work on the EPMIS/FAMIS system has reached a critical point. Management needs to establish a clear set of goals and priorities for the system development to succeed. My specific recommendations are:

1. Establish a set of goals for the system.

2. Design decision modules to implement these goals.

3. Design a database to support the decision modules.

4. Give contractors more guidance in pursuit of these goals. Booz, Allen and Hamilton have produced little useful in the EPMIS database design. Until they do, all development on both EPMIS and FAMIS is effectively blocked.

5. Improve the FAMIS prototype for short term demonstration purposes in the following ways.

   a) Compile the FAMIS prototype.

   b) Move to a hard disk environment.

   c) Use on faster micro.

   d) Speed up the graphics.

   e) Use as a test bed for future enhancements.

6. Design an administrative support structure for FAMIS.

   a) Define goals for integrated EPMIS/FAMIS.

   b) Develop administrative procedures for data management.

   c) Try to achieve high degree of interoperability between EPMIS and FAMIS.

d) Plan to operate in a permanent prototyping mode. This means that a capable systems team should be able to make fast changes to the system without going through a whole systems development cycle.

e) Develop a software team used to quick response and development.

7. Implement the following hardware and software steps.

a) Choose compatible software for EPMIS/FAMIS.

b) Design for interoperability between EPMIS VAX system and FAMIS micro system.

c) The best system choices are the C language for custom applications, SQL for database and FOCUS for unstructured queries into the database.

d) Replace Otrona with hard disk 80286 based portable.

e) Plan to replace the replacement in a few years.

f) Encrypt the FAMIS database.

g) Watch developments in laser disks.

Future Research Considerations

Much needs to be done to make EPMIS/FAMIS a truly useful system. We do not have the resources needed to do everything required. That will take years. I have outlined the steps NPS can accomplish in the next year.

1. Design a data dictionary structure for capturing data about the database.

2. Produce an initial FAMIS database design.

3. Produce a compiled C version of the current FAMIS file based prototype.

4. Produce a "proof of concept" prototype of a database oriented set of FAMIS routines using an 80286 based computer, a hard disk system, C language and SQL database management system.

5. Produce a "proof of concept" prototype of high resolution graphics routines for improving FAMIS geodata support.

6. Produce a database system that is capable of passing NUDET geodata to FAMIS in latitude and longitude format from input in location, direction and offset format.

CONTENTS

1

INTRODUCTION

The Office of the Manager of the National Communication System (NCS) requires timely accurate information about communications resources during national disasters. This led to the realization that an automated decision support system (DSS) could be useful to the National Emergency Management teams. These teams will play control the monitoring and resolution of claims on the system.

Meeting these requirements is the task of the integrated emergency preparedness management information system (EPMIS) and the fly-away management system (FAMIS). They are an integral part of the national telecommunications management system. This system will be an integrated system of mainframe and microcomputers. The mainframe computers contain the information necessary to reconstruct the national communication system. Analysts in the field use microcomputers to process the information that they will need in the reconstruction of the system.

The system is still in the early design stages. This report covers mostly the FAMIS portion of the system. Of necessity, however, it also includes material on the integration with EPMIS. The FAMIS system currently exists as a prototype. This prototype was written in BASIC and implemented on portable Otrona microcomputers. The present system is a decision support system of the file drawer variety. It is used solely for information retrieval. Eventually, information analysis and expert system capabilities will be necessary.

The FAMIS prototype is obviously in an early stage of development. Many enhancements are needed to make it a feasible system for production use. The prototype has been a useful learning tool for defining the final system. The prototype has generated interest in the project.

Now that the concept has been approved it is time to move beyond the prototype stage. A disciplined development effort is required. The remainder of this report deals with the definition of the problem and the process of setting up a disciplined development effort.

2

CURRENT FAMIS CAPABILITIES

The FAMIS microcomputer systems is a 2,700 line BASIC program. It is written in GW-BASIC, an MS-DOS dialect. This is because the system uses the Otrona portable microcomputer. GW-BASIC is similar to IBM's advanced BASIC. Judging from the dates on the disks we received, development stopped in late 1983 or early 1984. Little documentation is available beyond one document from NCS [National Communications System, August 1983] which listed the menus and brief descriptions of their functions.

The best way to examine the system is to look at the menus. These show the functions and what they do. The main menu for the system is below.

FAMIS Main Menu

1. P.O.C. Lists
2. Emergency Activation Procedures
3. Network Status Monitoring
4. Damage Assessment
5. Resolution of Claim
6. Zooming
7. Word Processing (WordStar)
8. NSC Processing Module

Item 1 is point of contact list. This option allows the user to get the primary point of contact for various government and private agencies. The program returns the point of contact name, address, phone number, and alternative means of contact. There is currently no formal means of updating these point of contact lists. It is possible to update the list using the word processing selection later in the menu. To do this effectively the user would have to know a lot about the data structures employed by the system. Even then it would be a risky process. There is too much chance that a novice would damage the database and render the system inoperable.

Item 2 is emergency activation procedures. This option displays instructions for activating the emergency procedure. It includes information on a conditions required before activation of the NCS. This option is a reference manual of procedures for the field analyst using FAMIS.

Item 3 is network status monitoring. Selection of this item generates graphic views of the status of government and commercial communications networks. This can be done at either the national or regional level. Map displays show connectivity and current damage status of major switching centers.

Item 4 is damage assessment. This selection allows the user to update the current network status data. The user may add information on observed or predicted damage to network switching centers or communication nodes.

Item 5 is resolution of claim. This option has not yet been implemented. Presumably it will allow the user to request that a particular node or communications link be reestablished or modified. This request is called a claim and all claims are held by the system and are available to be listed for the user. A major task of the NCS is to assign a priority to claims on the system.

Item 6 is zooming. This feature was not provided in the material we received from NCS.

Item 7 is word processing. The word processing is not built into the FAMIS system. Word processing is implemented by calling an external word processor. Here the system calls WordStar. This feature will not work on the IBM advanced BASIC version 2.0. It requires the SHELL command which does not exist in IBM's advanced BASIC 2.0. This feature is in IBM's BASIC 3.0. With this selection, a user can edit external files. It is used to modify the database or compose messages.

The final menu item is the NSC processing module. Its purpose was not explained in the documentation we received. Presumably it gives contacts on the National Security Council. Perhaps the data is too sensitive for general circulation.

Using a prototype has enabled NCS to field a version of the FAMIS system faster than using a traditional systems life cycle. The prototype is similar to the final product. It has been useful to generate interest in the project.

Nevertheless, the prototype has many problems. Its usefulness is at an end. The system was written in BASIC, an interpreted language. This allowed for fast development because compiling and link editing were unnecessary. This is the only good news about BASIC. Everything else is bad news. Because it is an interpreted language, it is slow. Non-programmers designed BASIC for use by non-programmers. It is presumably easy to learn. That may be true but it is also hard to use. The language is unstructured. Programs in BASIC are extremely difficult to maintain. Data handling is slow and clumsy. It does not lend itself to sophisticated file and data base techniques common to modern languages like C or PASCAL.

As is usual in a prototype, there are many errors in the system. There are occasional unexplained abends. Many of the features present in the menus are not yet implemented. Some of the features present are incorrect. This is true in the the map oriented systems. In the damage assessment modules, map displays do not work properly. A rectangular damage area fails to plot properly. The code needs to be thoroughly rewritten and tested. I suspect the map uses a conical projection and the rectangular damage areas use a Mercator projection. Clearly the system needs a more sophisticated geodata display.

There are additional problems with this system. It is largely undocumented. The only documentation is a listing of screen displays and menus. The program itself is mostly uncommented. This may be necessary since BASIC is an interpreted language. By using few comments, the interpreter does not waste time processing comments. But, it makes the resulting product nearly impossible to maintain.

Because of the limited length of BASIC variable names, it is difficult to design data names for mnemonic value. This makes it hard for maintenance programmers to determine what is going on with data handling. Finally, the program uses several questionable practices to accomplish its ends. One example is the menu item that allows the user to exit to WordStar. This gives the user free range to change values in data files and fields. The system is complicated enough for an experienced programmer or data analyst. One would not want to turn an inexperienced user loose on these data files. The result can only be disaster.

2.1 ANALYSIS STEPS TAKEN

In analyzing this system I had two major goals in mind. The first was to speed up the existing system for demo purposes. The second goal was to analyze the system to determine the features or changes needed for future revisions.

I took several steps to see if I could speed it up. They included:

1. Rewriting portions of the system.

2. Moving the system to a hard disk.

3. Developing techniques to speed up the graphics.

4. Compiling a portion of the code.

5. Running a compiled system using an 80186 based accelerator board.

6. Rewriting a portion of the code in C.

7. Invesitgating data base management systems for both EPMIS and FAMIS.

These approaches offered some help. Moving the system to a hard disk was useful. The major benefit was that the user no longer had to worry about swapping floppy disks. The necessary files were automatically loaded off the hard disk. There was a slight increase in speed (perhaps by a factor of 2). But, the system was so slow to begin with that this could hardly be called major. Compiling the system using a Microsoft BASIC compiler helped much more. Unfortunately the Microsoft BASIC compiler is limited to 64K. I could not compile the complete system. I was able to compile the main driver module and the damage assessment module. The damage assesment module contains the most detailed graphics. I figured that this is an important routine that compilation should speed up. The compiled version of the code runs somewhere between 5 and 10 times faster than the floppy disk version. This is good but still not acceptable.

Next I tried to speed up the graphics. The NCS programs perform graphics a point at a time using the BASIC graphics commands. This means that it may take as much as a minute to generate a map of the United States and communications facilities. It may not sound like much but it is painfully slow sitting in front of a computer. Even in interpreted BASIC it is possible to achieve speeds much faster than this. This is done by using the BSAVE and BLOAD commands. This makes it possible to generate a screen in about three seconds.

The same screen graphics capabilities are available in other languages. Using C, it is possible to generate screens in about a tenth of a second. The disadvantage of this approach is that the entire screen must be saved on a file. A graphics screen requires 16K of storage space. This might be a problem on a floppy disk based system. On a hard disk system it is no problem at all. Since hard disk systems are cheap, future designs for FAMIS should assume hard disk or laser disk storage. There is no point in locking into the older technology.

Running the compiled version of FAMIS using an 80186 based accelerator board produced dramatic increases in speed. Menus and graphics are generated quickly. The FAMIS system makes heavy demands on a microcomputer. Future versions of this system should use faster chips than the old 8088 technology. The same comment applies here as applied to disks. Design the system using that best technology available.

Rewriting a portion of the code in C gave the biggest increase in speed. A structured language like C or PASCAL is simply better suited to this kind of problem. In addition, C offers much greater availability of sophisticated program libraries than BASIC. These libraries have functions for screen formatting, graphics and data base.

The final step was the investigation of data base management systems (DBMS). In the prototype, a change to a data structure requires a change to the program. The major benefit of a DBMS is that it allows program-data independence. One can change file structures without reprogramming. There are several DBMS that are interoperable on minis and micros, Unfortunately, none of them work with BASIC.

2.2 CONCLUSIONS

For the short run, there are several things that can be done to make the prototype more useful. They are:

1. Compile it.

2. Move its programs and data to a hard disk.

3. Run it on a faster processor.

4. Use stored graphics geodata.

This means the Otrona microcomputers have outlived their usefulness. A more useful computer would be a Compaq with a hard disk. Unfortunately, this computer will be be obsolete within a year or two. They

will be replaced by portable systems using faster chips, larger hard disks, and laser disks. All this is normal in such fast moving technology.

Running the prototype on a more powerful machine would merely be a cosmetic change. It is the technical eqivalent of putting a Model T engine in a Porsche body. We can make the prototype run faster, but it cannot be easily expanded or modified. BASIC has too many faults for a production version of FAMIS. We should take our lessons from the BASIC prototype and move on.

The drivers for the screens and graphics should use a compiler language. The data structure should be implemented on a compatible data base management system (DBMS). The system should use a language and data base system that is interoperable with the mainframe system. This would allow a system that is more powerful, more flexible, and more reliable than the current prototype.

3

DEFINING DECISION MODULES

The steps required are:

1. Review and if necessary, revise the goals of the combined EPMIS/FAMIS systems.

2. Design the decision modules needed to support the goals.

3. Identify the data needed to support the decision modules.

4. Select the hardware and software configuration needed to support the system.

5. Begin work on the system using a disciplined prototyping approach rather than the full systems life cycle approach.

The first step should be to identify the decision and data components of the decision support system (DSS). The most important investment in the project so far is not the BASIC prototype program. It is the data and the sources of data that are used by the prototype. If the managers of the project do not take steps soon to create a formal data structure and data gathering system, this intellectual capital will be lost.

An important step in this process is data design. It is necessary to identify the data needed to support decisions made by the system. It is also necessary to develop models and decision aids needed by the analysts to reconstruct the communications system. Once we define the data structure, we need to design the data distribution. This also includes designing networks for communications between the center and field analysts.

The issue of hardware and software presents a dilemma for the EPMIS/FAMIS system design. Typically, hardware and software selection are done after the data design and decision design components of the system are completed. But, hardware and software for this type of DSS is changing rapidly. Any decisions made now are likely to be obsolete in a short time. The decision requirements will change constantly as well. Managers must realize that this system will never be complete. It will always be evolving.

There are many developments coming up soon that will impact EPMIS/FAMIS design. Examples include high quality laser disc systems, faster microcomputers, better graphics, and expert systems. The most important thing is to create a flexible logical design for the system. We can find hardware and software to carry out the logical design. If we concentrate first on hardware and software design then the logical design issues are likely to be lost.

It is necessary to set up an administrative infrastructure to support the operation. I would place the offices of data administrator and of systems administrator at the top of the list. The data administrator acts as the custodian and keeper of the data in the system. Much of the data is highly volatile. Information is included on points of contact and procedures. These change rapidly and is the responsibility of the data administrator to track these changes and enter them into the system.

This means that NCS must create a data collection system. It is necessary to identify the sources of data and to keep the data current. One might use techniques such as a computerized tickler file to send out letters to monitor changes in the system. The data in the prototype is out of date. This is not surprising since it is about two years old. If 20% of the points of contact change every year (probably low), the data has a three year half-life.

The responsiblity of the data administrator is to track this data and see that it is current. The responsibility of the systems administrator is to keep abreast of technical developments that affect the system. This responsibility includes monitoring the design of both the EPMIS and FAMIS sides of the system. This means tracking technical developments that would affect either system. The systems administrator is in charge of the configuration and development of the system.

The hardware and software in the systems must be easy to maintain and technically current. The software and data structures should be be usable across machine lines. This is not easy, but micros and minis are evolving so that it is possible. As microcomputers become more sophisticated, they share many systems with their larger cousins. A design goal of the EPMIS/FAMIS system should be to use the same programs and data structures on both systems.

4

CHOOSING SOFTWARE

The hardware and software decisions must support the overall system goals. The EPMIS/FAMIS managers must not fall into the trap of the traditional systems development life cycle. The life cycle model views a computer system as a solution to a problem. We encounter a problem. We analyze it. We design a system to solve it. Finally, we implement that system on a computer.

This may work well enough when working with simple, bounded problems. These are not the problems that the EPMIS/FAMIS system is intended to solve. These systems are decision aids for an unstructured enviroment. System requirements will continue to evolve over the life of the system. If we try to state them fully at the beginning, we will never finish. Daniel McCracken and Michael Jackson [1982, p. 30] stated the problem best.

> The life cycle concept perpetuates our failures so far, as an industry, to build an effective bridge across the communications gap between end-users and systems analysts. In many ways it constrains future thinking to fit the mold created in response to failures of the past.

In this system, we should make our software decisions first. Then, we can find hardware on which to implement this software. The software goals should be:

1. The software should be portable across machine lines. It should be useable on both micros and minis.
2. The software should be reasonably fast. The user should not have to wait forever for text or graphics output.

3. The software should support a DBMS. We must design our applications so that data and programs are independent. This is a major goal of a DBMS. If programs and data are not independent, maintenance and enhancement is almost impossible.

4. The language chosen for the software should be a structured language suitable for modern programming practices.

5. The software should support queries by untrained users.

This is a formidable set of requirements. The language chosen must allow for fast development. It will have to support sophisticated applications like graphics and data base. At the same time it will have to support untrained users.

To choose a language, we must examine the choices available on both microcomputers and minicomputers. The first language we will examine is BASIC. BASIC stand for Beginners All-Purpose Symbolic Instruction Code. It was invented at Dartmouth in 1964 and it evolved continuously since then. It is widely available on many classes of computers. The advantages and disadvantages of BASIC are listed below.

BASIC - Advantages

1. BASIC is somewhat standardized.
   The implementations available on microcomputers and minicomputers have many features in common. Programmers could move from one environment to the other without having to relearn the language.

2. BASIC is easy to learn.
   Most people can write simple programs in a few hours. In a few weeks they are able to do complicated programs.

3. BASIC is an interpreted language.
   This means that it is fast to write and change programs.

4. Compilers are available for BASIC.
   The interpreted programs can be translated into object modules which run much faster than the interpreted version.

5. Programmers in BASIC are widely available.
   Most programmers know BASIC.

BASIC-Disadvantages

1. BASIC is slow to execute.
   A large program in interpreted BASIC can be painfully slow. Even compiled BASIC is slower than compiled versions of more sophisticated languages.

2. BASIC is unstructured.
   It does not support constructs like the IF-THEN-ELSE, the DO-WHILE or the PERFORM-UNTIL. Some versions of BASIC have these constructs. But, these versions of BASIC sacrifice portability. Because it is unstructured, BASIC encourages the worst possible programming habits. Programs written in BASIC are nearly impossible to maintain.

3. BASIC is hard to use.
   This sounds paradoxical since an advantage of BASIC is that it is easy to learn. Easy to learn does not mean easy to use. Programs written in BASIC quickly become unmanageable. A programmer quickly loses control of the program. When the size of a program goes much above a hundred lines, there is little hope of cleaning it up.

4. BASIC's data handling features are poor.
   Beginning students seldom deal with large amounts of data. This may not be a handicap for teaching, but it is a handicap for use with the EPMIS/FAMIS systems. BASIC's lack of a record structure makes it extremely difficult to handle data.

5. BASIC does not support data base management systems.
   It is either difficult or impossible to link from BASIC to DBMS available on the same machines.

The next choice of language is COBOL. COBOL stands for Common Business Oriented Language. It was defined by a joint committee of users and Department of Defense in 1959. COBOL is widely used for data processing in both government and the private sector. The advantages and disadvantages of COBOL are listed below.

COBOL-Advantages

1. COBOL is a standardized language.
   The versions available on many different systems are similar.

2. COBOL is a data processing language.

It is ideally suited for handling fields and records. It has a wide variety of editing functions built in.

3. There are many programmers available who know COBOL.
   It is not as widespread as BASIC, but it is the language of choice in the commercial programming world. It is not difficult to find programmers familiar with this language.

COBOL-Disadvantages

1. COBOL is an old fashioned language.
   It does not support some of the modern control structures. It is also difficult to break a COBOL program up into modules or subroutines.

2. COBOL does not support data base well.
   It handles data well in a file processing environment. The links to a data base may or may not be available.

3. COBOL is slow.

4. COBOL has a poor to non-existent mathematics capability.
   If your problems require any numeric processing, COBOL is a poor choice.

5. COBOL is intended for batch processing operations.
   It supports screen-oriented on-line operations poorly.

FORTRAN is the next language to consider. FORTRAN stands for Formula Translator. It is an old language, developed at IBM in the 1950's. It is since become the standard language for scientific processing. It is available on minicomputers, mainframes and micros. The various versions of FORTRAN are usually quite compatible. It is an easy language to transfer across machines. In spite of this, FORTRAN is uncommon on microcomputers. The advantages and disadvantages of FORTRAN are summarized below.

FORTRAN-Advantages

1. FORTRAN is a standardized language that moves easily from machine to machine.

2. There are many programmers available for FORTRAN.
   Until recently, FORTRAN was the first language that most people learned. It is since been supplanted by BASIC.

3. FORTRAN has excellent mathematical processing capabilities.
   There are many mathematical libraries available for FORTRAN.

4. FORTRAN is well suited to minicomputers and mainframes.

FORTRAN-Disadvantages

1. FORTRAN is an old fashioned language.
   The older versions of it are the most transportable, and they do not support structured programming constructs.

2. FORTRAN has poor data handling features.
   It does not support record structures at all. String manipulation in FORTRAN can be maddening.

3. FORTRAN usually does not support interface to DBMS.

4. FORTRAN, like COBOL, is not well suited for screen oriented operations.

The next language, C, was developed at the Bell Telephone Laboratories in the late 1960's and early 1970's. It is a simple, yet powerful, structured programming language. It allows users to perform systems programming activities without having to resort to assembly language. The advantages and disadvantages of C are summarized below.

C - Advantages

1. The various versions of C are standardized and stable.
   The language was invented to facilitate transfer of programs from one type of machine to another. It performs this function well.

2. C is a structured language.
   It has the full range of structured constructs.

3. C has powerful features.
   These range from record oriented constructs at the top level down to constructs that allow you to manipulate bits and bytes.

4. There is a wide choice of support libraries available for C.
   These include libraries for mathematics, graphics, screen handling and language extensions.

5. C is the language of choice on microcomputers and minicomputers.
   Many of the systems available on microcomputers are written in C. Much the same is true on minicomputers. On Digital Equipment Corporation's PDP-11 and VAX systems, C is the language used to implement the UNIX operating system.

6. C offers interfaces to DBMS.
   Many of the DBMS available on microcomputers are written in C.

7. Interpreters are available for C.

C-Disadvantages

1. C programmers are hard to find.
   A recent issue of the Government Computer News noted that C programmers in the Washington, D.C. area frequently receive five figure raises when changing jobs. This is because C is the language of choice for implementing military command and control systems and sophisticated systems software. Programmers of this type are in demand.

2. C is not a language for beginners.
   It requires sophistication and skill in programming.

The next language is PASCAL. PASCAL was developed by Nicholas Wirth in Zurich, Switzerland in the 1970's. It is named after Blaise Pascal, the French mathematician. PASCAL is intended to be an instructional language. It encourages structured programming habits. The language has grown in popularity since its invention. It now is the language of choice for beginning instruction in Computer Science. It has replaced BASIC for serious computing instruction. It is widely popular on microcomputers and minicomputers. The advantages and disadvantages of PASCAL are summarized below.

PASCAL-Advantages

1. PASCAL is somewhat standardized. It is widely available on mainframes, minicomputers and microcomputers. The versions on different machines differ somewhat. This can be annoying but, generally, it is not serious.

2. PASCAL is a structured language and offers a wide variety of powerful features.

3. The language is widely available on mini and microcomputers.

4. PASCAL programmers are readily available.
Most computer science students from good programs have done their work in PASCAL.

PASCAL-Disadvantages

1. PASCAL has a narrower range of support packages than is available in C.

2. There are usually no special interfaces to DBMS in PASCAL, although one could probably modify C interface techniques to work.

3. There are no PASCAL interpreters available.
This may be less of a problem than it seems. The most popular version of PASCAL on microcomputers is Borland's Turbo-PASCAL. Turbo is a compiler but it operates fast enough that it is as easy to use as an interpreter.

4. PASCAL is slower than C because of the strong typing of variables used to support the instructional mission of the language.

5. The versions of PASCAL that are available on microcomputers do not support the large memory models.
They think of the machine as a 64K machine and cannot use the 1 megabyte and larger memories on newer machines.

The next language is FOCUS. This is a language developed in the 1970's by a company called Information Builders Inc. It one of a class of languages called Fourth Generation Languages (4GL) or Higher Order Languages (HOL) [Harel and McLean, 1985; Martin, 1982]. These are integrated languages that pre-dated packages like Symphony and Framework by 10 to 15 years. These languages allow novices to use integrated data base management systems, spreadsheet modeling, statistical packages and graphics.

Recently, the vendors of these languages have begun to offer their products on microcomputers as well as mainframe computers. One design goal in their product is to facilitate data and program transfer from microcomputers to mainframe and minicomputers. They offer the same version of their languages in both environments. The advantages and disadvantages of FOCUS are summarized below.

FOCUS - Advantages

1. FOCUS is an integrated system available on both micro and minicomputers.

2. FOCUS is easy to learn.

3. FOCUS supports ad hoc queries into the data base.

4. FOCUS supports a powerful data processing programming capability.

FOCUS-Disadvantages

1. The system is large and clumsy. On a microcomputer it requires 512K of memory and 3 and a half megabytes of the hard disk.

2. FOCUS is slow.

3. FOCUS is business data processing oriented.
The geodata routines required by the EPMIS and FAMIS will have to be programmed in C.

5

CONCLUSIONS

Work on the EPMIS/FAMIS system has reached a critical point. NCS has created structures to gather data about the communications systems. It has made a start on a system to store this data in computer readable form. Much remains to be done, however. Management needs to establish a clear set of goals and priorities for the system development to succeed. Currently, the goals of the project are unclear. The computer systems supporting the project reflect this. EPMIS/FAMIS system cannot succeed until NCS has a clear vision of its mission. It is impossible to design a decision support system without a clear idea of the decisions supported.

In one sense, this may never be possible. EPMIS and FAMIS gather information and support decisions in an unstructured, chaotic environment. There can never be a finished system. The EPMIS/FAMIS project will always have to operate in a prototyping mode.

NCS needs to assemble a flexible programming team to implement this system. This system team must also be highly competent. They should be able to design implement new programming and data structures quickly. In the conditions under which EPMIS/FAMIS will operate, there will be no time for a complete system development life cycle. A primary design goal must be flexibility. The specific recommendations for the EPMIS/FAMIS system follow.

First, establish specific goals for the system. The EPMIS/FAMIS system as it now stands is simply a file drawer system. It stores information about the current status of the communications system. FAMIS operators can enter status changes to that system. The system needs to be able to do more than this.

The NCS managers must develop means for setting priorities and resolving claims on the communications system. EPMIS/FAMIS should offer support for these decisions. This issue is politically sensitive, but until it is resolved, NCS cannot be effective in an actual crisis situation. NCS must have well defined goals to implement an effective decision support system.

Next, it is necessary to design decision modules to implement the goals. Until the goals are clear, the nature of the decision modules will not be clear either. The simple file drawer decision support system has its uses. But, more sophisticated decision support modules are needed. Examples might include mathematical programming algorithms to help reestablish connectivity. Another possibility would be queueing or simulation models to help predict actual load on the system. Expert system decision modules would be of immense help in codifying the rules used to resolve claims on the communications system.

Finally, begin design of a database to support the decision modules. The current system is not a database. It is a file oriented system. File oriented systems were common in the third generation computer languages. They caused the structure of the data to be hopelessly imbedded in programs. If the data structure is changed, then programs must be changed as well. As the system grows, this becomes an increasingly complex task. The result is that once the system has reached a certain size, it is too much trouble to change it. The system can no longer expand. Given the goals of the EPMIS/FAMIS system, this would be disaster.

The solution is a database management system. The best choice would be a relational database management system. Relational database is the newest technology and the easiest to work with. Database gives you program-data independence. It allows expansion of the database to take place independently of the programs.

Work on data design should begin immediately. Data design can take place before choosing a database management system. Eventually a data design must be implemented on a specific database management system. Design issues are independent of the database management system (for relational systems), so the specific relational system chosen is irrelevant. The first step in data design is to establish a data dictionary structure to capture facts about the data base. Facts include field names, field lengths, field types and sources of data.

The design process should use the current FAMIS files. These files are the most valuable part of the FAMIS prototype. They serve as a guide in moving from a prototype system to a production system. Begin immediately to isolate the data items, to assign them reasonable field names, and to begin the data dictionary building.

Once the data dictionary system is established, we can begin to assign these fields to files or relations. The relations should be in third normal form. This allows for easy modification of the data later. Once the data structure has been laid down, NCS can set up an administrative structure for data capture. The data that makes up the FAMIS

system has a short half life. This half life is almost certainly less than three years. There must be a means of capturing information about changes to the system.

The data dictionary system will serve as a basis for tracking changes. The process of tracking changes should be automated to the greatest extent possible. Using the FAMIS data dictionary system, one could set up a computerized letter generator. This system could periodically request updated information from the relevant agencies. The EPMIS/FAMIS system is only as good as the data making it up.

As part of this project, the material from Booz, Allen and Hamilton was reviewed. Booz, Allen and Hamilton has produced little useful material for EPMIS/FAMIS so far. Their reports are mostly rehashes of standard material. They are only marginally useful in the design of the EPMIS/FAMIS system. One report is largely a program listing of the FAMIS prototype. It has a few comments about the program, but it offers nothing new or useful.

Two other reports are rewrites of standard system analysis texts (the book by Powers, Adams and Mills, 1984 comes to mind as a possible source, but many sources are possible). The material has been rewritten to apply to the NCS situation. There is nothing wrong with using the best available sources in a report. The only problem with this material is that it is too superficial and too general. Booz, Allen and Hamilton should have provided a detailed analysis of the NCS situation.

My hypothesis is that Booz, Allen and Hamilton have not received sufficient guidance. The project needs to be better defined. What they have produced is a reflection of the current state of the EPMIS/FAMIS system. They seem to feel that the FAMIS system could remain as it is, written in BASIC and implemented on the Otrona microcomputers. Considering the needs of the FAMIS system, this is a strange recommendation. There are obviously problems with FAMIS. Booz, Allen and Hamilton should be the first to point them out. I doubt that they have assigned their first team to this project. Their work looks like a term paper in an MBA Systems Analysis class. To use high-priced consulting talent, one must provide well-defined tasks with explicit deliverables.

The next recommendations deal with improvement of the FAMIS prototype. The prototype is near the end of its life cycle. It is useful in the short term for demonstration and training, but it is not a production system. Several steps can be taken to enhance its usefulness. These include:

1. Compile the system.

   It is possible to speed the system up by a factor of between 10 and 20 by compiling it. The compiler used should be able to use all available memory. Such compilers include the new Microsoft 3.0 version of BASIC or an outside compiler like the Better BASIC compiler.

2. Move the system to a hard disk.

   The current system uses floppy disks. It is as much a test of the manual dexterity of the operator as anything else. The continual changing of floppy disks is an annoyance. We have put the system on a hard disk at the Naval Postgraduate School. The move was simple. All that was required was modification of the drive reference on open commands.

3. Implement the system on a faster microcomputer.

   An 80186 or an 80286 based machine offers a noticable improvement in speed. This improvement is even better when a compiled version of the prototype is used. These chips offer three to four times the speed of the conventional 8088. The 80286 or 80386 will probably be the target chip for the eventual FAMIS system. One may as well start working with them now.

4. Speed up the graphics on the system.

   The graphics on the FAMIS system are painfully slow. They are generated a point at a time using the BASIC graphics commands. Even in interpreted BASIC, it is possible to achieve faster speeds than this. One can use the BSAVE or BLOAD commands. This means that a graph that requires 20 seconds to generate in interpreted BASIC is cut to 3 seconds. If one uses machine language, the improvement is even more dramatic. The three seconds required by the BASIC commands can be cut to one-tenth of a second or less. These techniques require the use of 16K of file space for each graphic screen saved. This might be a severe limitation on a floppy disk based system. On a hard disk system it is not a problem.

5. Use the prototype system as a test bed for future enhancements, but ultimately abandon it.

   The prototype has outlived its usefulness and the sooner that its lessons can be moved to a more powerful system, the better.

The next point concerns the administration of the EPMIS/FAMIS systems. Management of the system is at least as important as the software and hardware. One needs to develop an administrative support structure. This includes both the data and the programming sides of the system. For the data side of the system it is necessary to develop an information resource management philosophy.

The first step should be to appoint a database administrator. The primary duty of the database administrator is to maintain configuration control on the database. It is this person's job to supervise the general structure of the system. The database administrator should oversee the management of the data using a data dictionary system. In this activity, the database administrator will have several subordinates. These would include clerks and data librarians. It is the function of the data librarian to supervise the data at the detail level. This includes such duties as making sure that the data is updated in a timely fashion.

The management of the system itself is as important as the management of the data. To give this system the flexibility it needs, NCS will have to have a staff of highly competent programmers. These individuals will have to be managed by a knowledgable systems manager. Given the way the federal government treats technical personnel, this may be an impossible dream. If this is true, then EPMIS/FAMIS is an impossible dream as well. It will be the responsibility of the system manager to maintain configuration control over the EPMIS/FAMIS system. This individual will have know the structure of the whole system and be able to supervise technical personnel.

A major goal of the EPMIS/FAMIS system should be a high degree of interoperability between the two systems. The EPMIS and FAMIS systems should use the same languages. This will reduce the cognitive load on the programmers working on the systems. It makes easier to transfer programs, data, and storage techniques. A few years ago, this would have been impossible. Now many of the same tools used on mainframes are being transported to micros. It is increasingly common to find the same programming languages used across machine lines.

The software team should plan to operate in a permanent prototyping mode. This will have to be a disciplined prototyping mode rather than the loose and sloppy operation that characterized the original FAMIS prototype. Programmers will have to learn to document as they go, to use structured programming techniques, and to work together as a team. This is a common approach among programmers operating in a UNIX environment on systems like C. It was obviously not the approach used in the original FAMIS prototype.

If the EPMIS/FAMIS system is to provide support in chaotic situations, this is the kind of team that is required. The software team will have to develop systems quickly. They will have to have a high level of technical sophistication. Given the environment in which FAMIS will operate, it is not possible to anticipate all demands on the system in advance. Those we can anticipate should be built into the system. For the others, it will be necessary to respond quickly to changing situations as they develop.

NCS needs to move quickly to lay a hardware and software foundation for the future. The decisions made now will determine the success of the EPMIS/FAMIS system. NCS should make some changes to the current hardware and software. First is to choose compatible software for EPMIS and FAMIS. The current software choices are C and Ingres for database for EPMIS, and BASIC and the IBM PC family for FAMIS. C is a reasonable language given the sophisticated requirements of this system. It operates on both minis and micros.

The choice of INGRES as a database management system for EPMIS also has problems. INGRES is a reasonable database management language, but it does not exist on the PC. The manufacturers of INGRES are planning to have a PC version sometime in 1986. Other database management systems have this option available now. Another disadvantage of INGRES is that it is not the direction in which database systems are moving. The main database development seems to be headed toward SQL.

There is a move afoot to create an ANSI standard for SQL. This language was invented at IBM and is available from several manufacturers. The best choices would be C for custom applications, SQL for database and perhaps FOCUS for applications requiring quick quick development. These packages are available on both mainframes and micros.

Once we have chosen compatible software, the managers of the system must design for interoperability between the VAX and micro system. The systems should use the same programs, database management system and data files. This will make for faster, cleaner, and cheaper implementation efforts.

A point must be mentioned about the FAMIS database. Strictly speaking, none of the data in the database is classified. But, it is not something that one would want to fall into unfriendly hands. FAMIS is an easy to use blueprint of the U.S. communications system. Moreover, in times war or national emergency, information about the status of communications systems would be classified. Some steps should be taken for protection of the of the FAMIS database. The database should be

encrypted. The data provided by a FAMIS operator could be used to decrypt the data.

The final point involves the computers used by FAMIS. The Otronas have outlived their usefulness. They were a good decision at the time, and now, they should be retired. Their replacement should be a hard disk based 80826 system. This would allow for greater storage capacities and speeds in the FAMIS system. A high resolution color monitor would also be desirable. The managers of the EPMIS/FAMIS system should watch developments in laser disks carefully. The geodata displays on a current FAMIS system are crude. The laser disks offer the possiblility of extremely sophisticated geodata capability. This would immeasurably enhance the effectiveness of the FAMIS system.

This report has outlined many steps that the managers of the EPMIS/FAMIS system should consider. The managers need to realize that both technical and administrative changes are required. There is no technical quick fix for the problems of NCS. A cleaner tighter administration is the first priority. Without that, technical changes would be of no benefit.

REFERENCES

Alter, Steven, **Decision Support Systems**, Addison-Wesley, New York, 1980.

Alter, Steven, "A Taxonomy of Decision Support Systems", **Sloan Management Review**, Vol. 19, No. 1, Fall 1981, pp. 39-56.

Appleton, D. S., "Data Driven Prototyping", **Datamation**, Vol. 29, November 1983.

Baroody, A. J. and DeWitt, D. J., "An Object-oriented Approach to Database System Implementation", **ACM Transactions on Database Systems**, Vol. 6, No. 4, 1981, pp. 576-601.

Batini, C., Lenzerni, M. and Santucci, G., "A Computer-aided Methodology for Conceptual Database Design", **Information Systems**, Vol. 7, No. 3, 1982, pp. 265-280.

Blum, B. I., "Rapid Prototyping of Information Management Systems", **ACM SIGSOFT Software Engineering Notes**, Vol. 7, p. 35, December 1982.

Blum, B. I., "Still More About Rapid Prototyping", **ACM SIGSOFT Software Engineering Notes**, Vol. 8, July 1983.

Boar, Bernard H., **Application Prototyping: A Requirements Definition Strategy for the 80's**, John Wiley and Sons, Inc., New York, 1984.

Booz, Allen & Hamilton Inc., "Emergency Preparedness Management Information System (EPMIS): System Specifications for Integrated EPMIS/FAMIS", Report NCS-0034-078, Booz, Allen & Hamilton, Inc., 4330 East West Highway, Bethesda, MD 20814, February 8, 1985.

Booz, Allen & Hamilton Inc., "Emergency Preparedness Management Information System (EPMIS): Software Requirements Specification", Report NCS-0034-087, Booz, Allen & Hamilton, Inc., 4330 East West Highway, Bethesda, MD 20814, June 28, 1985.

Booz, Allen & Hamilton Inc., "Emergency Preparedness Management Information System (EPMIS): Draft FAMIS Data Base Specifications", Report NCS-0034-093, Booz, Allen & Hamilton, Inc., 4330 East West Highway, Bethesda, MD 20814, July 15, 1985.

Brooks, Fredrick P., **The Mythical Man-Month**, Addison-Wesley Publishing Co., Inc., Reading, Mass., 1975.

Carey, T. T. and Mason, R. E. A., "Prototyping Interactive Information Systems", **Communications of the ACM**, Vol. 26, p. 347, May 1983.

Carlson, Eric D., et. al., "The Design and Evaluation of an Interactive Geo-Data Analysis and Display System", **Proceedings of the IFIPS Congress 74**, North-Holland Publishing Company, 1980, pp. 463-468.

Carlson, Eric D. (ed.),"Proceedings of a Conference on Decision Support Systems", **Database**, Vol. 8, No. 3, Winter 1977.

Carlson, Eric, "An Approach for Designing Decision Support Systems", **Database**, Vol. 10, No 3, Winter 1979, pp. 3-15.

Chamberlin, D.D., Astrahan, M. M., Eswaran, K. P., Griffiths, P. P., Lorie, R. A., Mehl, J. W., Reisner, P., and Wade, B. W., "SEQUEL 2: A Unified Approach to Data Definition, Manipulation and Control", **IBM Journal of Research and Development**, November 1976, pp. 560-575.

Chen, P. P., "The Entity-Relationship Model -- Toward a Unified View of Data", **ACM Transactions on Database Systems**, Vol. 1, No. 1, 1976, pp. 9-36.

Chen, P. P. (ed.), **The Entity-Relationship Approach to Systems Analysis and Design**, North-Holland, Amsterdam, 1981.

U. S. Dept. of Commerce, National Bureau of Standards, FIPS Pub. 46, Data Encryption Standard, January 1977.

U.S. Department of Commerce, National Bureau of Standards, NBS, FIPS PUB 83, Guidelines on User Authentication Techniques for Computer Network Access Control, Sep. 1980.

Department of Commerce, National Bureau of Standards, Special Publication 500-33, Subject: Considerations in the Selection of Security Measures for Automatic Data Processing Systems, June 1983.

Connell, John and Brice, Linda, "Practical Quality Assurance", **Datamation**, Vol. 31, No. 5, March 1, 1985, pp. 106-114.

Cooper, John D. and Fisher, Matthew J. (eds.), **Software Quality Management**, Petrocelli Books, Inc., 1979.

Couger, J. Daniel, Colter, Mel A. and Knapp, Robert W., **Advanced System Development/Feasibility Techniques**, John Wiley & Sons, Inc., New York, 1982.

Daly, Edmund B., "Organizing for Successful Software Development", **Datamation**, 25, 14, (1979), 106-120.

Davis, Randall, "Expert Systems: Where are we? And where do we go from here?", MIT AI Laboratory AI Memo No. 665, June 1982.

Davis, Michael W., "Anatomy of Decision Support", **Datamation**, pp. 201-208, 15 June 1984.

Denning, Dorothy E. and Denning, Peter J., "Data Security", **Computing Surveys**, Vol. 11, No. 3, September 1979, pp. 227-250.

Faurer, L.D., "Information Protection in the Federal and Private Sectors", **Computer Security Journal**, Vol. 2, No. 2, pp. 89-96, Fall/Winter 1983.

Faurer, L.D., "Computer Security Goals of the Department of Defense", **Computer Security Journal**, Vol. 3, No. 1, pp. 55-60, Summer 1984.

Flavin, M., **Fundamental Concepts of Information Modeling**, Yourdon Press, New York, 1981.

Fleet Material Support Office, CDA Development Handbook, FMSOINTINST 5230.20A, Mechanicsburg, PA, 1 December 79.

Francella, Kevin, "Information Resource Management: A Brief Overview", **Data Management**, January 1983.

Gillenson, Mark L., "The State of Practice of Data Administration", **Communications of the ACM**, Vol. 25, No. 10, October 1982, pp. 699-706.

Glaser, George, "Managing Projects in the Computer Industry", **Computer**, Vol. 17, No. 10, October 1984, pp 45-53.

Guides, E., "The Design of a Cryptography Based Secure File System", **IEEE Transactions on Software Engineering**, Vol. SE-6, No. 5, September 1980.

Guimares, Tor, "A Study of Application Program Development Techniques", **Communications of the ACM**, Vol. 28, No. 5, May 1985, pp. 494-499.

Guimares, Tor, "IRM Revisited", **Datamation**, Vol. 31, No. 5, March 1, 1985, pp. 130-134.

Guynes, S., Laney, M.G., and Zant, R., "Computer Security Practice", **Journal of Systems Management**, Vol. 34, No. 6, p. 22, June 1983.

Hammer, M. and McLeod, "Database Description with SDM: A Semantic Database Model", **ACM Transactions on Database Systems**, Vol. 6, No. 3, pp. 351-386.

Hanson, Owen, **Design of Computer Data Files**, Computer Science Press, 1982.

Harel, Elie C. and McLean Ephraim R., "The Effects of Using a Nonprocedural Computer Language on Programmer Productivity", **MIS Quarterly**, Vol. 9, No. 2, June 1985, pp. 109-120.

Harmon, Paul and King, David, **Expert Systems**, John Wiley and Sons, Inc., New York, 1985.

Hawryszkiewwycz, I. T., **Database Analysis and Design**, Science Research Associates, Chicago, 1984.

Hayes-Roth, Fredrick, "The Knowledge-Based Expert System: A Tutorial", **Computer**, Vol. 17, No. 9, September 1984, pp. 11-28.

Hayes-Roth, Fredrick, "Knowledge-Based Expert Systems", **Computer**, Vol. 17, No. 10, October 1984, pp. 263-273.

Howden, William E., "Contemporary Software Development Environments", **Communications of the ACM**, Vol. 25, No. 5, May 1982, pp. 318-329.

Huber, George P., "Issues in the Design of Group Decision Support Systems", **MIS Quarterly**, pp. 195-204, September 1984.

Jackson, M. A. and McCracken, D. D., "Life Cycle Considered Harmful", **ACM SIGSOFT Software Engineering Notes**, Vol. 7, p. 31, April 1982.

Keen, Peter G. W., and Michael S. Scott Morton, **Decision Support Systems: An Organizational Perspective**, Reading MA: Addison-Wesley Publishing Company, 1978.

Keen, P. G. and Wagner, G. R., "DSS: An Executive Mind-Support System", **Datamation**, Vol. 25, No. 12, November 1979, pp. 117-122.

Keen, Peter G. W. "Decision Support Systems: Translating Analytic Techniques into Useful Tools", **Sloan Management Review**, Spring 1980, pp. 33-44.

Keen, Peter G. W., "Information Systems and Organizational Change", **Communications of the ACM**, Vol. 24, No. 1, January 1981, pp. 24-33.

Keen, Peter G. W., "Value Analysis: Justifying Decision Support Systems", **MIS Quarterly**, March 1981, pp 1-15.

Kent, William, "A Simple Guide to Five Normal Forms in Relational Database Theory", **Communications of the ACM**, Vol. 26, No. 2, pp. 120-125.

Kroenke, David M., **Database Processing**, Science Research Associates, 1983.

Landwehr, Carl E., "Formal Models for Computer Security", **Computing Surveys**, Vol. 13, No. 3, September 1981, pp. 247-278.

Landwehr, C.E., "The Best Available Technologies for Computer Security" **Computer**, Vol. 16, No. 7, July 1983.

Martin, James, **Application Development Without Programmers**, Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1982.

Martin, James and McClure, C., **Software Maintenance: The Problem and Its Solution**, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1983.

McCririck, Ian B. and Goldstein, Robert C., "What Do Data Administrators Really Do?", **Datamation**, Vol. 26, No. 8, August 1980, pp. 131-134.

McNurlin, B. C., "Developing Systems by Prototyping", **EDP Analyzer**, v. 19, September 1981.

Metzger, Phillip, **Managing a Programming Project**, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1973.

Michie, D, (ed.), **Introductory Readings in Expert Systems**, Gordon and Breach Science Publishers, 1982.

Michaelsen, Robert and Michie, Donald, "Expert Systems in Business", **Datamation**, Vol. 29, No. 11, November 1983, pp. 240-246.

Mills, John A., "A Pragmatic View of the Systems Architect", Communications of the ACM, Vol. 28, No. 7, July 1985, pp. 708-717.

Mintzberg, Henry, "Managerial Work: Analysis from Observation", Management Science, Vol. 18, No. 3, October 1971, pp. B-97 - B-112.

Mintzberg, Harvey, The Nature of Managerial Work, Harper and Row, New York, 1973.

Murray, W.H., "Good Computer Security Practices for Two Areas of Current Concern: Personal Computers and Dial-up Systems", Computer Security Journal, Vol II, No. 2 pp. 77-88, Fall/Winter 1983.

Nau, Dana S., "Expert Computer Systems", IEEE Computer, Vol. 16, No. 2, February 1983, pp. 63-85.

Naumann, Justus D., and A. Milton Jenkins. "Protyping: The New Paradigm for Systems Development", MIS Quarterly, September 1982, pp. 29-44.

National Communications System, "Emergency Telecommunications Micro Subsystem", report prepared by National Communications System, Washington, D.C. 20305, August 1983.

Ng, P. A., "Further Analysis of the Entity-Relationship Approach to Database Design", IEEE Transactions on Software Engineering, Vol. SE-7, No. 1, 1981, pp. 85-99.

Parker, D.B., Computer Security Management, Reston Publishing Company, Inc, 1981.

Parkin, A., "Data Analysis and System Design by Entity-Relationship Modelling", Computer Journal, Vol. 25, No. 4, 1982, pp. 401-409.

Parker, D.B., "The Many Faces of Data Vulnerability", IEEE Spectrum, Vol. 21, No. 5, May 1984.

Powers, Michael J., Adams, David R., and Mills, Harlan D., Computer Information Systems Development: Analysis and Design, South-Western Publishing, 1984.

Riemann, Bernard C. and Waren, Allan D., "User-Oriented Criteria for the Selection of DSS Software", Communications of the Association for Computing Machinery, Vol. 28, No. 2, February 1985, pp. 166-179.

Rockart, John, "Chief executives define their own data needs", Harvard **Business Review**, Vol. 57, No. 2, March - April 1979, pp. 81-92.

Ross, Douglas T., "Applications and Extensions of SADT", **Computer**, Vol. 18, No. 4, April 1985, pp. 25-35.

Rothnie, J. B., Bernstein, P. A., Fox, S., Goodman, N., Hammer, M., Landers, T. A., Reeve, C., Shipman, D. W., and Wong, E, "Introduction to a System for Distributed Databases (SDD-1)", ACM **Transactions on Database Systems**, Vol. 5, No. 1, March 1980, pp. 1-17.

Seaman, J., "Halting Network Intruders", **Computer Decisions**, Vol 17, 2, pp. 82-93, 29 January 1985.

Serlin, Omri, "Departmental Computing: A Choice of Strategies", **Datamation**, Vol. 31, No. 9, May 1, 1985, pp. 86-96.

Shanker, K.S., "The Total Computer Security Problem", in **Advances in Computer System Security**, Edited by Rein Turn, Artech House, Inc., 1981.

Silverman, M.E., "Selling Security to Senior Management, DP Personnel, and Users", **Computer Security Journal**, Vol II, 2, pp. 7-18, Fall/Winter 1983.

Smith, Henry C., "Database Design: Composing Fully Normalized Tables From a Rigorous Dependency Diagram", **Communications of the ACM**, Vol. 28, No. 8, August 1985, pp. 826-838.

Sprague, Ralph H., Jr., and Eric D. Carlson, **Building Effective Decision Support Systems**, Englewood Cliffs NJ: Prentice-Hall, Inc., 1982.

Stefik, Mark, Atkins, Jan, Balzer, Robert, Benoit, John, Birnbaum, Lawrence, Hayes-Roth, Fredrick, and Sacerdoti, Earl, "The Organization of Expert Systems, A Tutorial", **Artificial Intelligence**, Vol. 18, 1982, pp. 135-173.

Stefik, Mark, "Strategic Computing at DARPA: Overview and Assessment", **Communications of the ACM**, Vol. 28, No. 7, July 1985, pp. 690-704.

Sweet, Frank, "Painting a New Picture", **Datamation**, Vol. 31, No. 10, May 15, 1985, pp. 132-140.

Sweet, Frank, "Managing Data-Driven Development", **Datamation**, Vol. 31, No. 12, June 15, 1985, pp. 131-138.

Synnott, W. R. and Gruber, W. H., **Information Resource Management**, John Wiley, 1981.

Teichroew, Daniel and Hershey, Ernest A., "PSL/PSA: A Computer-Aided Technique for Structured Documentation and Analysis of Information Processing Systems", in Couger, et. al., **Advanced System Development/Feasibility Techniques**, 1982.

Teichroew, Daniel, Hershey, Ernest A. and Yamamato, Y., "The PSL/PSA Approach to Computer-Aided Analysis and Documentation", in Couger, et. al., **Advanced System Development/Feasibility Techniques**, 1982.

Teorey, Toby J., and Fry, James P., **Design of Database Structures**, Prentice-Hall, Englewood Cliffs, NJ, 1982.

Thayer, R. H., Pyster, A. B., and Wood, R. C., "Major Issues in Software Engineering Project Management", **IEEE Transactions on Software Engineering**, SE-7, 4, (1981), 333-342.

Tonies, Charles C., "Project Management Fundamentals", in Jensen and Tonies, **Software Engineering**, 1979.

Vetter, M. and Maddison, R. N., **Database Design Methodology**, Prentice-Hall International, London, 1981.

Voydock, V.L., and Kent, S.T., "Security Mechanisms in High Level Network Protocols", **ACM Computing Surveys**, Vol 15, 2, pp. 135-171, June 1983.

Wasserman, A. I. and Shewmake, D. T., "Rapid Prototyping of Interactive Information Systems", **ACM SIGSOFT Software Engineering Notes**, v. 7, December 1982.

Wasserman, A. I., **Introduction to System Design Methodology, Tutorial on Software Design**, IEEE Computer Society Press, 1983.

Weldon, J. L., **Data Base Administration**, Plenum Press, 1981.

Wetherbe, James C., **Systems Analysis and Design**, West Publishing Company, 1984.

Wiederhold, Gio, "Databases", **Computer**, Vol. 17, No. 10, October 1984, pp. 211-223.

Wood, C.C., "Countering Unauthorized Systems Access", **Journal of Systems Management**, Vol 35, No 4, p. 26, April 1984.

Yourdon, Edward **Managing the Structured Techniques**, Second Edition, Prentice-Hall, Englewood Cliffs, New Jersey, 1979.

Zaniolo, C. and Melkanoff, M. A., "On the Design of Relational Database Schemata", **ACM Transactions on Database Systems**, Vol. 6, No. 1, 1981, pp. 1-47.

Zaniolo, C. and Melkanoff, M. A., "A Formal Approach to the Definition and Design of Conceptual Schemata for Database Systems", **ACM Transactions on Database Systems**, Vol. 7, No. 1, 1982, pp. 24-58.

Distribution List

Recipient	No. Copies
Director of Research Code 014 Naval Postgraduate School Monterey, CA 93943	1
Defense Technical Information Center Cameron Station Alexandria, VA 22314	2
Library Code 1424 Naval Postgraduate School Monterey, CA 93943	2
Professor Norman R. Lyons Code 54LB Naval Postgraduate School Monterey, CA 93943	10
Mr. Edward Cain NCS/PP 8th Street and South Courthouse Road Arlington, VA 22204	2
Mr. Ken Boheim NCS/PP 8th Street and South Courthouse Road Arlington, VA 22204	1
Dr. Bruce Barrow NCS/PP 8th Street and South Courthouse Road Arlington, VA 22204	1
Mr. Norman Douglas NCS/EP 8th Street and South Courthouse Road Arlington, VA 22204	1
COL William Schooler NCS/EP 8th Street and South Courthouse Road Arlington, VA 22204	1

LTC Tom Cindric 1
JDSSC
8th Street and South Courthouse Road
Arlington, VA 22204

Professor M. G. Sovereign 1
Code 55ZO
Naval Postgraduate School
Monterey, CA 93943

Professor James R. Yee 1
Code 55YE
Naval Postgraduate School
Monterey, CA 93943

Professor Carl R. Jones 1
Code 54JS
Naval Postgraduate School
Monterey, CA 93943

Professor Jack W. La Patra 1
Code 54LP
Naval Postgraduate School
Monterey, CA 93943

Professor Norman F. Schneidewind 1
Code 54SS
Naval Postgraduate School
Monterey, CA 93943

Professor Tung Bui 1
Code 54BD
Naval Postgraduate School
Monterey, CA 93943

Professor Dan Dolk 1
Code 54DK
Naval Postgraduate School
Monterey, CA 93943

Professor Taracad Sivasankaran 1
Code 54SE
Naval Postgraduate School
Monterey, CA 93943